

```

int MPI_Send(void* buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)
int MPI_Recv(void* buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

int MPI_Isend(void* buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm, MPI_Request *request)
int MPI_Irecv(void* buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Request *request)
int MPI_Wait(MPI_Request *request, MPI_Status *status)

```

FIG. 1

Wrapper	Event Record
<pre> extern int MPI_Send(void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm) { int res = 0; INT64 duration=0; u_TIMESTRUCT start,end; u_GETTIME(&start); res = PMPI_Send(buf, count, datatype, dest, tag, comm); u_GETTIME(&end); duration = u_GETTIMEDIFF(&end,&start); /* package event structure and write it to memory buffer */ return res; } </pre>	<pre> <1082> MPI_Send { int64 timestamp[1]; int32 rank[1]; int64 duration[1]; int32 count[1]; int32 datatype[1]; int32 dest[1]; int32 tag[1]; int32 comm[1]; } { 1082 6123339879 1 716 1250 20 0 2000 0 } { 1082 6123400004 3 636 1250 20 2 2000 0 } { 1082 612341445 0 343 1250 20 1 3000 0 } { 1082 612341472 2 339 1250 20 3 3000 0 } </pre>

FIG. 5

MPI Task 0	MPI Task 1
<pre> #define size 1024 int sdata[size]; int rdata[size]; MPI_Status status; MPI_Request request; int tag = 30; /* initialization */ /* ---- blocking send-recv 0 -> 1 */ /* fill sdata */ MPI_Send (sdata, size, MPI_INT, 1, tag, MPI_COMM_WORLD); /* use or overwrite sdata */ /* ---- non-blocking send recv 1->0 */ MPI_Irecv (rdata, size, MPI_INT, 1, tag, MPI_COMM_WORLD, &request); /* computation excluding rdata */ MPI_Wait(&request,&status); /* use rdata */ /* finish */ </pre>	<pre> #define size 1024 int sdata[size]; int rdata[size]; MPI_Status status; MPI_Request request; int tag = 30; /* initialization */ /* ---- blocking send-recv 0 -> 1 */ MPI_Recv (rdata, size, MPI_INT, 0, tag, MPI_COMM_WORLD, &status) /* use rdata */ /* ---- non-blocking send recv 1->0 */ /* fill sdata */ MPI_Isend (sdata, size, MPI_INT, 0, tag, MPI_COMM_WORLD, &request); /* computation excluding sdata */ MPI_Wait(&request,&status); /* use or overwrite sdata */ /* finish */ </pre>

FIG. 2

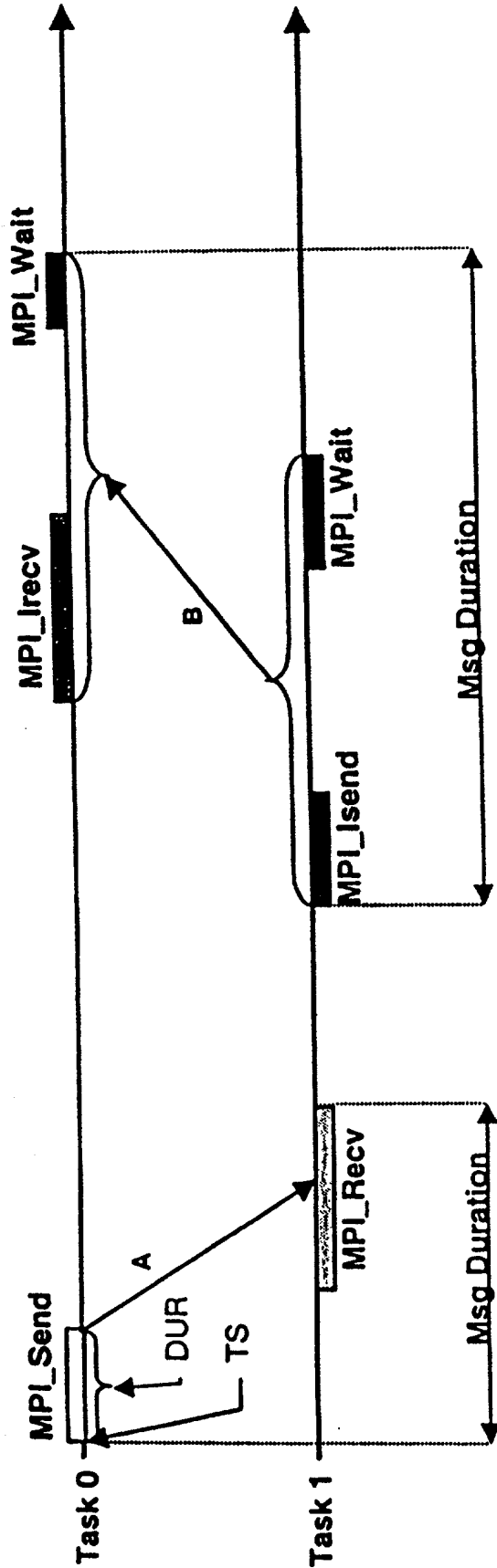


FIG. 3 A

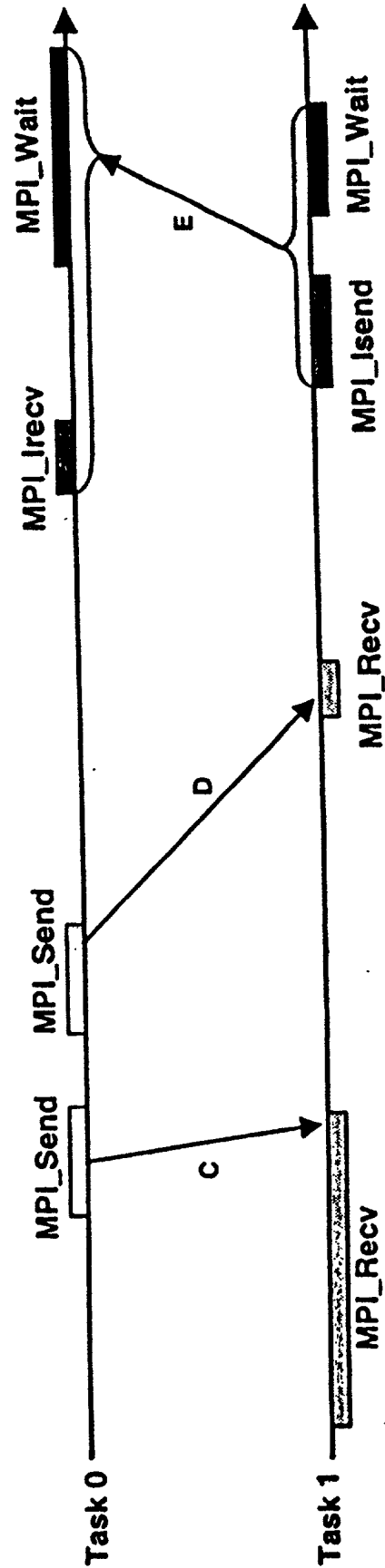


FIG. 3 B

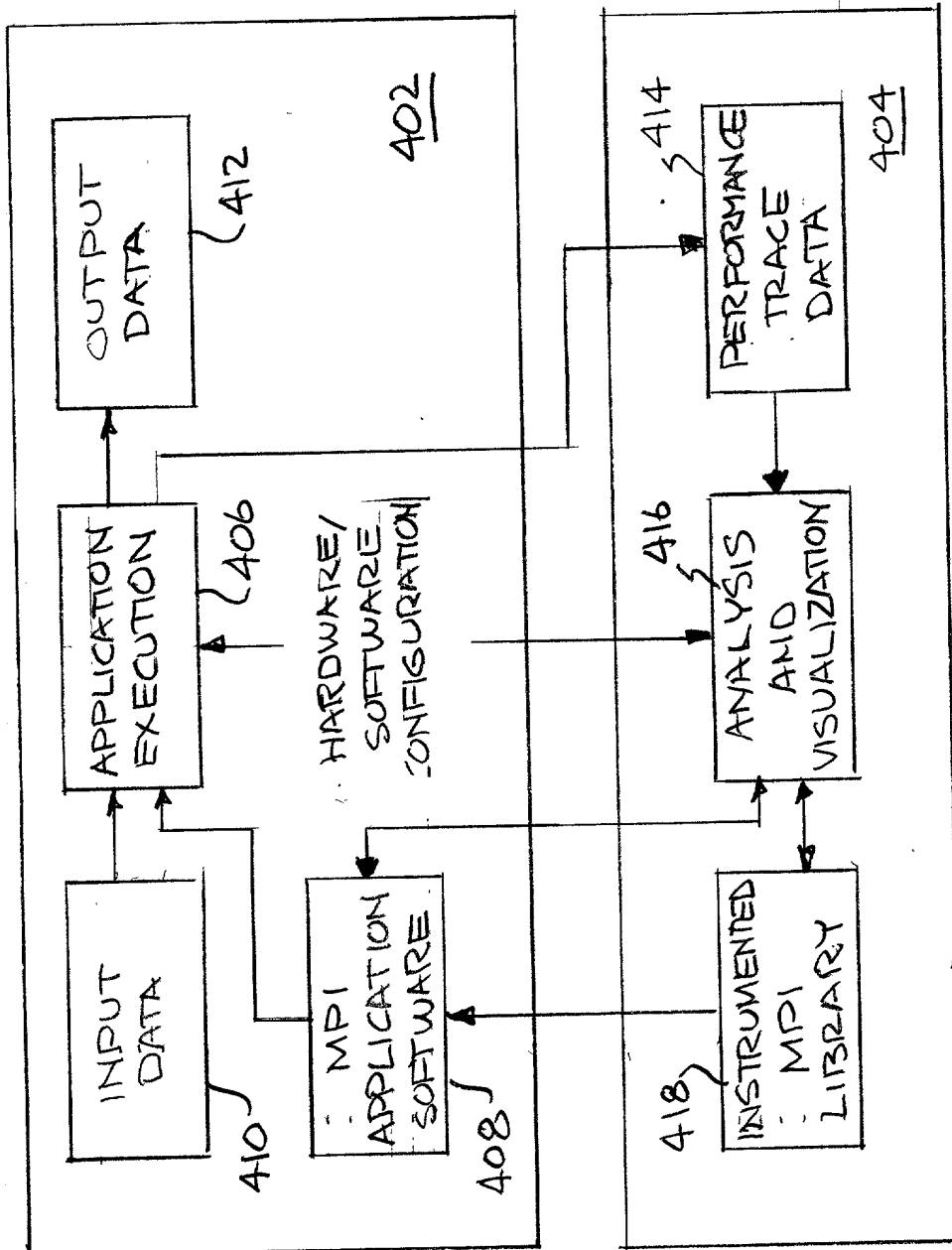


FIG. 4

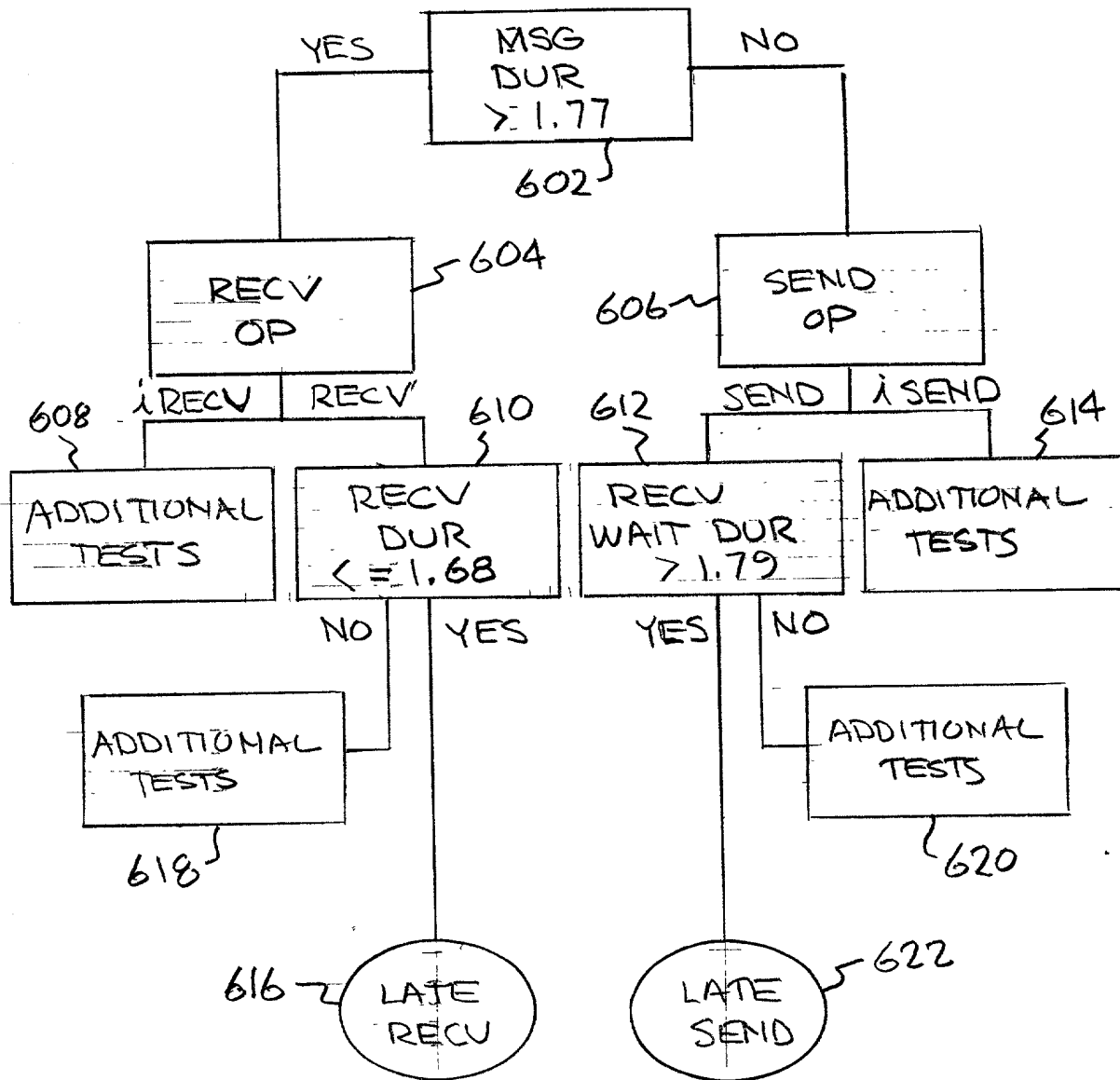


FIG. 6

6/10

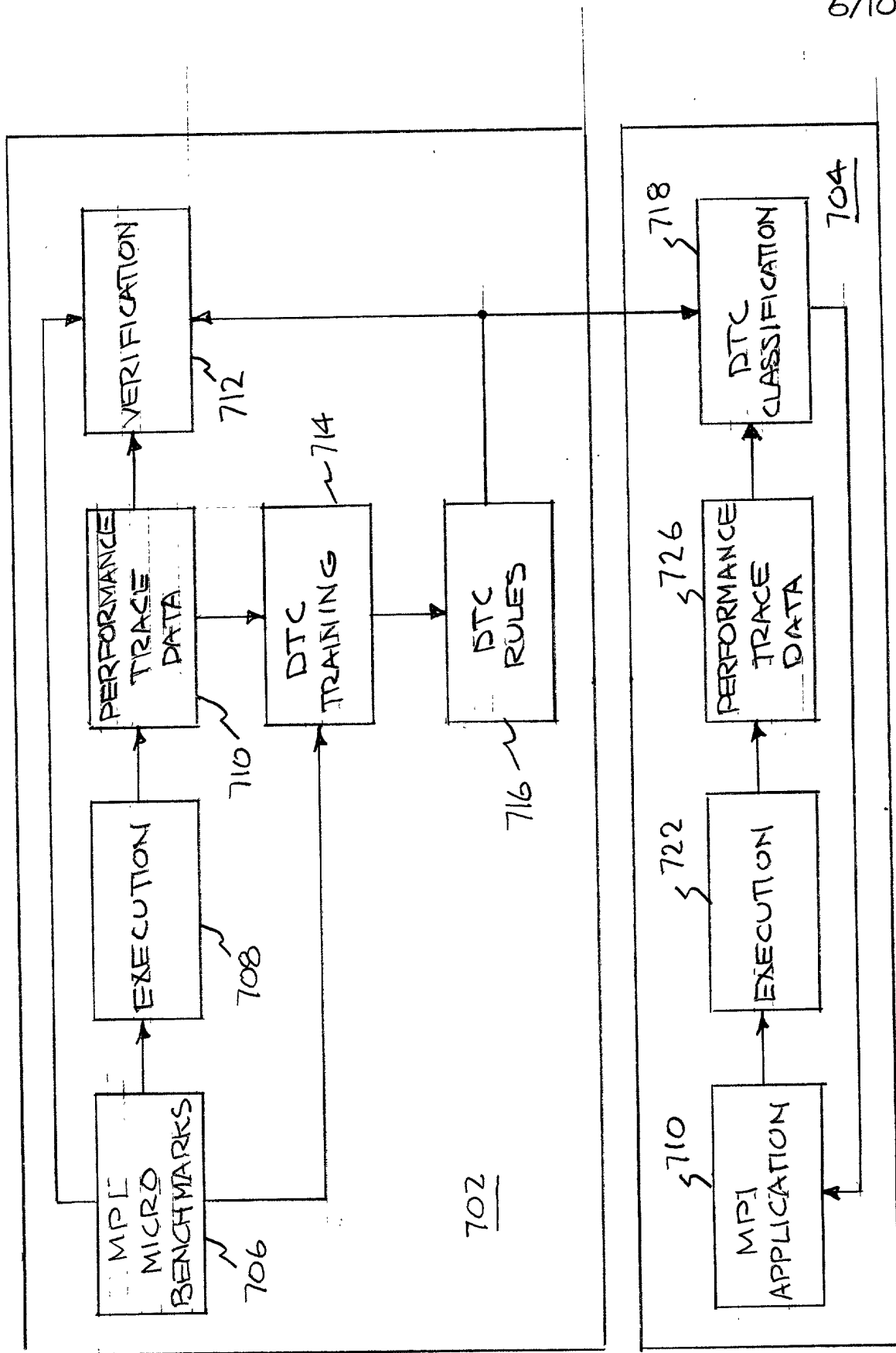


FIG. 7

Send Type	Send Dur	Send Wait Dur	Recv Type	Recv Dur	Recv Wait Dur	Msg Size	Msg Dur	Class
send	1.06	0.00	recv	0.98	0.00	16	0.97	normal
send	1.00	0.00	recv	0.92	0.00	16	0.93	normal
isend	1.77	1.22	recv	41.49	0.00	2	0.70	late send post
isend	1.26	1.14	recv	4.50	0.00	2	0.73	late send post
isend	0.65	1.84	irecv	1.12	0.82	32768	1.83	late recv post
isend	0.93	3.15	irecv	1.29	0.87	32768	3.05	late recv post

FIG. 8

Prediction							
	normal	late send	late rcv	late rcv post	late rcv wait	late send post	late send wait
normal	786	4	3	3	2	14	28
latesend	13	406	0	0	0	0	0
laterecv	21	0	399	0	0	0	0
laterecvpost	48	1	0	314	55	0	2
laterecvwait	48	0	0	127	245	0	1
latesendpost	20	0	0	0	0	397	3
latesendwait	16	0	1	0	0	9	394

FIG. 9

Send Type	Send Dur	Send Wait	Recv Type	Recv Dur	Recv Wait	Msg Size	Msg Dur	Class	Confidence Factor
send	2.90	0.00	irecv	2.70	1.86	1250	2.07	late recv post	0.92
send	3.35	0.00	irecv	1.21	1.60	1250	2.36	late recv wait	0.99
send	3.36	0.00	irecv	1.21	1.45	1250	2.34	late recv wait	0.99

FIG. 10

Location		Class						
Sender	Receiver	Normal	Late Send	Late Recv	Late Send Post	Late Send Wait	Late Recv Post	Late Recv Wait
		2	0	0	34	0	0	1572
		0	0	0	1600	0	0	4
copy_faces+4268	copy_faces+3980							
x_solve+7716	x_solve+888							

FIG. 11

Application	Trace File Size (MB)	Number of Trace Records	Normal Runtime (seconds)	MPI_Send	MPI_Recv	MPI_Isend	MPI_Irecv
CG-HEAT	2.7	39698	70	6828	148	0	6680
NAS BT	5.6	88522	1344	0	0	9672	9672
NAS SP	1.6	31394	840	0	0	19272	19272
sPPM	0.2	2066	287	0	0	480	480

FIG. 12

IL-10,620 S-94,206

10/10

Application	Sender Location	Receiver Location	Class						
			normal	late send	late rcv	late send post	late send wait	late rcv post	late rcv wait
NAS BT	copy_faces+4056	copy_faces+3636	795	0	0	2	0	11	0
	copy_faces+4000	copy_faces+3696	0	0	0	755	0	0	53
	copy_faces+4168	copy_faces+3756	2	0	0	22	0	0	784
	copy_faces+4112	copy_faces+3816	3	0	0	6	0	0	799
	copy_faces+4280	copy_faces+3876	4	0	0	35	0	0	769
	copy_faces+4224	copy_faces+3936	3	0	0	27	0	0	778
	x_solve+612	x_solve+124	804	0	0	0	0	0	0
	x_solve+432	x_solve+120	0	0	0	28	162	0	614
	y_solve+608	y_solve+124	804	0	0	0	0	0	0
	y_solve+420	y_solve+120	0	0	0	35	208	0	561
	z_solve+908	z_solve+124	804	0	0	0	0	0	0
	z_solve+348	z_solve+120	0	0	0	21	183	0	600
9672			3219	0	0	931	553	11	4958
NAS SP	copy_faces+4100	copy_faces+3680	1575	0	0	6	0	27	0
	copy_faces+4044	copy_faces+3740	0	0	0	1474	0	1	133
	copy_faces+4212	copy_faces+3800	6	0	0	22	0	0	1580
	copy_faces+4156	copy_faces+3860	3	0	0	17	0	0	1588
	copy_faces+4324	copy_faces+3920	0	0	0	33	0	0	1575
	copy_faces+4268	copy_faces+3980	2	0	0	34	0	0	1572
	x_solve+7716	x_solve+888	0	0	0	1600	0	0	4
	x_solve+17864	x_solve+8272	0	0	0	1604	0	0	0
	y_solve+7632	y_solve+872	0	0	0	1602	0	0	2
	y_solve+17236	y_solve+8180	0	0	0	1604	0	0	0
	z_solve+7612	z_solve+912	0	0	0	1603	0	0	1
	z_solve+17224	z_solve+8184	0	0	0	1604	0	0	0
19272			1586	0	0	11203	0	28	6455
CG HEAT	snd_int+80	rcv_int+128	141	3	4	0	0	0	0
	snd_r8+80	rcv_async_r8+136	1396	2321	0	4	0	819	-2140
6828			1537	2324	4	4	0	819	2140
SPPM	xbdry+1248	xbdry+152	40	0	0	0	0	0	0
	xbdry+1184	xbdry+212	40	0	0	0	0	0	0
	xbdry+6028	xbdry+288	29	0	0	0	10	0	1
	xbdry+5960	xbdry+356	6	0	0	12	12	0	10
	ybdry+1248	ybdry+152	40	0	0	0	0	0	0
	ybdry+1184	ybdry+212	40	0	0	0	0	0	0
	ybdry+6588	ybdry+288	31	0	0	0	9	0	0
	ybdry+6520	ybdry+356	8	0	0	10	9	0	13
	zbdry+6268	zbdry+288	19	0	0	0	19	0	2
	zbdry+6200	zbdry+356	14	0	0	16	5	0	5
	zbdry+10920	zbdry+432	33	0	0	0	7	0	0
	zbdry+10852	zbdry+500	12	0	0	17	4	0	7
480			312	0	0	55	75	0	38

FIG.13